Aula 04 - Circuitos lógicos combinacionais, circuitos lógicos MSI e circuitos aritméticos

Hugo Vinícius Leão e Silva

hugovlsilva@gmail.com, hugo.vinicius.16@gmail.com, hugovinicius@ifg.edu.br

Instituto Federal de Educação, Ciência e Tecnologia de Goiás
Campus Anápolis
Curso de Bacharelado em Ciência da Computação

10 de setembro de 2021

Visão geral

Formas: soma-de-produtos e produto-de-somas

- Circuitos combinacionais →a saída do circuito está em função da combinação das entradas;
- Para simplificá-los, eles devem estar representados usando:
  - **Soma-de-produtos**:
    $ABC + \overline{A}B\overline{C}$
    $AB + \overline{A}B\overline{C} + \overline{C}\,\overline{D} + D$
    $ABC + \overline{A}B + C\overline{D} + EF + GK + H\overline{L}$
  - **Produto-de-somas**:
    $(A + \overline{B} + C)(A + C)$
    $(A + \overline{B})(\overline{C} + D)F$
    $(A + C)(B + \overline{D})(\overline{B} + C(A + \overline{D} + \overline{E}))$

- Simplifique:

$$
\begin{aligned}
X &= ABC + A\overline{B}\,\overline{\overline{A}\,\overline{C}} \\
&= ABC + A\overline{B}\,(\overline{\overline{A}} + \overline{\overline{C}}) \\
&= ABC + A\overline{B}\,(A + C) \\
&= ABC + A\overline{B}A + A\overline{B}C \\
&= ABC + A\overline{B} + A\overline{B}C \quad \text{(soma-de-produtos)} \\
&= AC(B + \overline{B}) + A\overline{B} \\
&= AC(1) + A\overline{B} \\
&= AC + A\overline{B} \\
&= A(C + \overline{B}) \quad \text{(produto-de-somas)}
\end{aligned}
$$

- Exemplos 4-2 a 4-6 (pp. 100-101)

Mapa de Karnaugh

- É um método gráfico para simplificar circuitos;
- Aplicabilidade prática para circuitos de até seis entradas;
- O **mapa K** relaciona as entradas e a saída do circuito:
  - Cada linha da tabela verdade corresponde a um quadrado no mapa K;
  - Os quadrados adjacentes diferem entre si apenas por uma variável;
  - A partir do mapa K preenchido com 0's e 1's, monta-se a expressão na forma de soma-de-produtos para o circuito.

Tabela: Montagem do mapa K para um circuito de duas variáveis

A	**B**	**X**
0	0	1 $\rightarrow \overline{A}\,\overline{B}$
0	1	0
1	0	0
1	1	1 $\rightarrow AB$

	$\overline{B}$	B
$\overline{A}$	1	0
A	0	1

Qual é a expressão algébrica para o circuito?

Mapa de Karnaugh - agrupando pares

A	B	C	X
0	0	0	1 $\rightarrow \overline{A}\,\overline{B}\,\overline{C}$
0	0	1	1 $\rightarrow \overline{A}\,\overline{B}C$
0	1	0	1 $\rightarrow \overline{A}B\overline{C}$
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	1 $\rightarrow AB\overline{C}$
1	1	1	0

	$\overline{C}$	C
$\overline{A}\,\overline{B}$	1	1
$\overline{A}B$	1	0
AB	1	0
$A\overline{B}$	0	0

Qual é a expressão algébrica para o circuito?
Pares de 1's adjacentes $\rightarrow$ elimina-se a variável acompanhada do seu complemento.
Figs. 4-12(a, b, c, d)

Mapa de Karnaugh - agrupando quartetos

	$\overline{C}$	C
$\overline{A}\,\overline{B}$	1	0
$\overline{A}\,B$	1	0
$A\,B$	1	0
$A\,\overline{B}$	1	0

	$\overline{C}\,\overline{D}$	$\overline{C}D$	CD	$C\overline{D}$
$\overline{A}\,\overline{B}$	0	1	1	0
$\overline{A}\,B$	0	1	1	0
$A\,B$	0	0	0	0
$A\,\overline{B}$	0	0	0	0

Qual é a expressão algébrica para o circuito?
Quartetos de 1's adjacentes $\rightarrow$ eliminam-se as duas variáveis acompanhadas do seu complemento.
Figs. 4-13(a, b, c, d, e)

Mapa de Karnaugh - agrupando octetos

	$\overline{C}\,\overline{D}$	$\overline{C}D$	CD	$C\overline{D}$
$\overline{A}\,\overline{B}$	1	1	1	1
$\overline{A}B$	0	0	0	0
AB	0	0	0	0
$A\overline{B}$	1	1	1	1

Qual é a expressão algébrica para o circuito?
Quartetos de 1's adjacentes $\rightarrow$ eliminam-se as três variáveis acompanhadas do seu complemento.
Figs. 4-14(a, b, c, d)

Mapa de Karnaugh - Exercícios práticos

	$\overline{C}\,\overline{D}$	$\overline{C}D$	CD	$C\overline{D}$
$\overline{A}\,\overline{B}$	0	0	0	1
$\overline{A}B$	0	0	0	0
AB	1	1	0	0
$A\overline{B}$	1	1	1	0

	$\overline{C}\,\overline{D}$	$\overline{C}D$	CD	$C\overline{D}$
$\overline{A}\,\overline{B}$	0	0	0	1
$\overline{A}B$	0	1	1	0
AB	0	1	1	0
$A\overline{B}$	0	1	1	0

	$\overline{C}\,\overline{D}$	$\overline{C}D$	CD	$C\overline{D}$
$\overline{A}\,\overline{B}$	0	0	1	0
$\overline{A}B$	1	1	1	1
AB	1	1	0	0
$A\overline{B}$	0	0	0	0

Mapa de Karnaugh - Exercícios práticos

	$\overline{C}\,\overline{D}$	$\overline{C}D$	CD	$C\overline{D}$
$\overline{A}\,\overline{B}$	0	1	0	0
$\overline{A}B$	0	1	1	1
AB	1	1	1	0
$A\overline{B}$	0	0	1	0

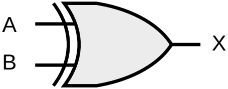
	$\overline{C}\,\overline{D}$	$\overline{C}D$	CD	$C\overline{D}$
$\overline{A}\,\overline{B}$	0	1	0	0
$\overline{A}B$	0	1	1	1
AB	0	0	0	1
$A\overline{B}$	1	1	0	1

	$\overline{C}$	C
$\overline{A}\,\overline{B}$	1	1
$\overline{A}B$	1	1
AB	0	0
$A\overline{B}$	0	1

Porta lógica XOR

- XOR $\rightarrow$ *Exclusive-OR*/OU-exclusivo;
- É representada pelo símbolo $\oplus$;
- $A \oplus B = B \oplus A$ (associatividade);
- $A \oplus (B \oplus C) = (A \oplus B) \oplus C$ (comutatividade);
- É possível fazer XOR de mais de duas entradas cascateando as portas;
- X $= 1$ sempre que a quantidade de 1's for ímpar.

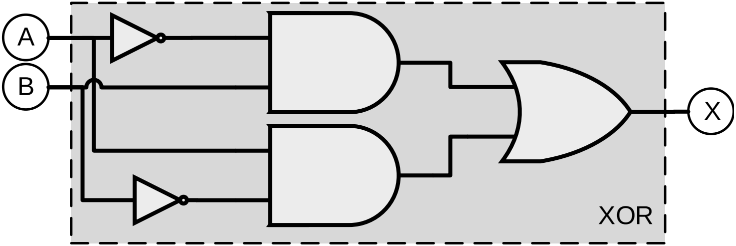
A	B	$X = A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0



Como é a tabela-verdade de $X = A \oplus B \oplus C$? E o seu circuito?

Porta lógica XOR

Figura: Implementação da porta XOR

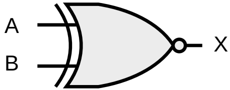


Qual é a expressão algébrica para este circuito? E para $X = A \oplus B \oplus C$?

Porta lógica `XNOR`

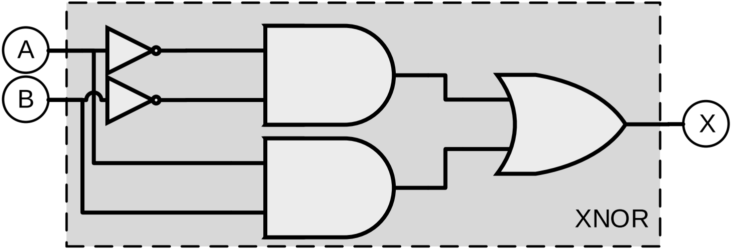
- `XOR` $\rightarrow$ *Exclusive-NOR*/função coincidência;
- É representada pelo símbolo $\odot$;
- `X` $= 1$ sempre que a quantidade de 1's for par.

A	**B**	$X = \overline{A \oplus B} = A \odot B$
0	0	1
0	1	0
1	0	0
1	1	1



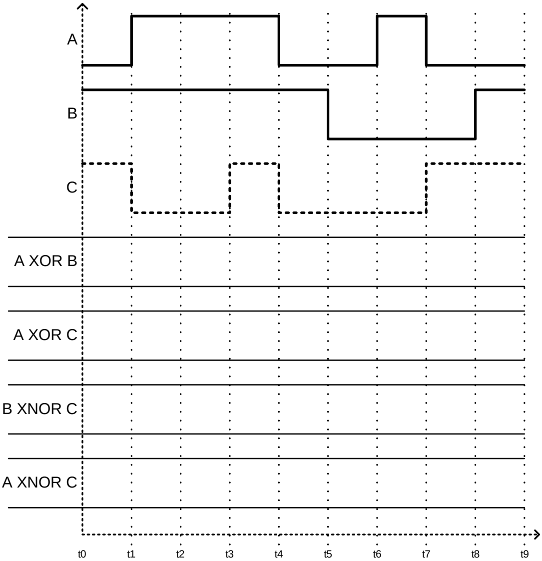
Porta lógica XNOR

Figura: Implementação da porta XNOR



Exemplos 4-16 e 4-17 (Pg. 116-Tocci)
Qual é a expressão algébrica para este circuito? E para $X = A \odot B \odot C$?

Exercício XOR e XNOR



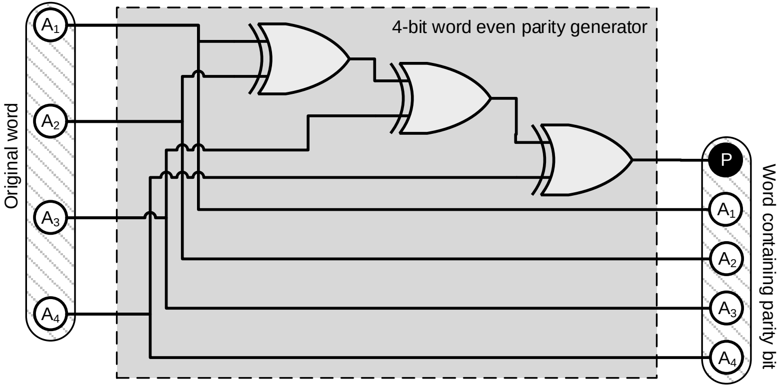
Paridade

Conta-se a quantidade de 1's. A palavra resultante deve ter quantidade ímpar ou par de 1's.

A	B	C	D	Paridade ímpar	Paridade par
0	0	0	0		
0	0	0	1		
0	0	1	0		
0	0	1	1		
0	1	0	0		
0	1	0	1		
0	1	1	0		
0	1	1	1		
1	0	0	0		
1	0	0	1		
1	0	1	0		
1	0	1	1		
1	1	0	0		
1	1	0	1		
1	1	1	0		
1	1	1	1		

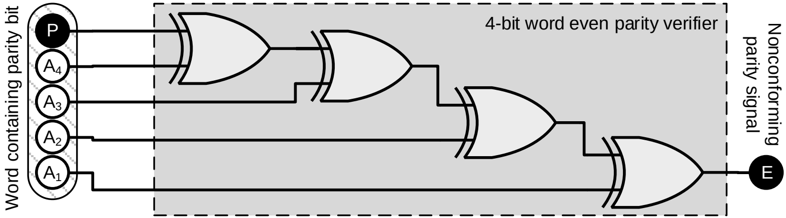
Gerador/verificador de paridade

Exemplo de um codificador utilizando paridade par para uma palavra de quatro bits:



Gerador/verificador de paridade

Exemplo de um verificador de paridade par para uma palavra de quatro bits:

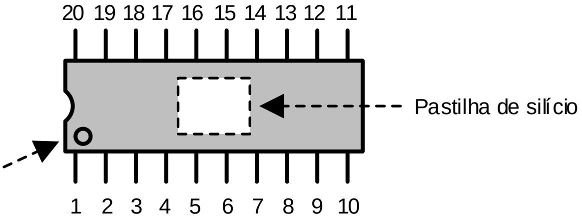


E = 0 – não-erro

E = 1 – erro

É possível saber qual bit está errado?

- **Lógica discreta** vs. **lógica de circuitos integrados** (CIs);
- CIs são compostos por resistores, diodos, transístores em um(a) única(o) pastilha/pedaço (aka *chip*) de silício encapsulada(o) em um substrato de plástico ou cerâmica com os pinos;
- Existem diversos tipos de encapsulamento, exemplos: DIP (*dual in-line package*), PGA (*pin grid array*) [ZIF – *zero insection force*], BGA (*ball grid array*) e LGA (*Land Grid Array*) [FC – *flip chip*].



Circuitos Integrados (CIs) e família de CIs

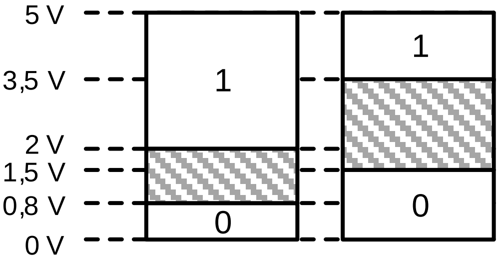
- Níveis de integração de circuitos:

Nível de integração	**N° de portas lógicas**
SSI (*Small scale integration*)	$n < 12$
MSI (*Medium scale integration*)	$12 \leq n \leq 99$
LSI (*Large scale integration*)	$100 \leq n \leq 9.999$
VLSI (*Very large scale integration*)	$10.000 \leq n \leq 99.999$
ULSI (*Ultra large scale integration*)	$100.000 \leq n \leq 999.999$
GSI (*Giga scale integration*)	$n \geq 1.000.000$

- **TTL** (*Transistor-Transistor Logic*) – utiliza transístores bipolares e é usada em CIs SSI e MSI
- **CMOS** (*Complementary Metal-Oxide Semiconductor*) – utiliza transístores de efeito de campo (MOSFET) e é utilizada em CIs LSI ou maiores

Cuidado ao combinar chips TTL e CMOS no mesmo circuito!

Figura: Níveis de tensão para CIs TTL e CMOS



Códigos

- **Códigos** são conjuntos organizados de símbolos/sinais apropriados para determinada aplicação:
  - Canal – códigos detectores ou corretores de erro;
  - Fonte – criptografia e compressão de dados;
  - Linha – nível elétrico para cada um dos símbolos do código.
- Exemplos: ASCII, Unicode, BCD, E<u>BCD</u>IC, RGB, **Gray** etc;

Códigos – Código Gray

Decimal	Binário	Código Gray
0	0000	0000
1	0001	0001
2	0010	0011
3	0011	0010
4	0100	0110
5	0101	0111
6	0110	0101
7	0111	0100
8	1000	1100
9	1001	1101
10	1010	1111
11	1011	1110
12	1100	1010
13	1101	1011
14	1110	1001
15	1111	1000

Códigos – Código Gray

- Código utilizado em MODEMs (moduladores/demoduladores) "analógicos";
- Muda apenas um bit de um símbolo para outro – muito semelhante ao Mapa de Karnaugh:

	00	01	11	10
00	0	7	8	15
01	1	6	9	14
11	2	5	10	13
10	3	4	11	12

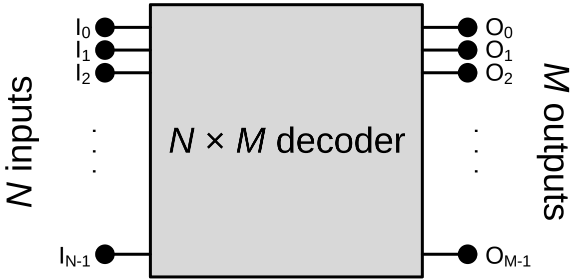
Códigos – Código 2 entre 5

- Sempre há dois bits '1' em uma palavra de cinco bits – codificação menos eficiente;
- Usado na telefonia e facilita a detecção de erro (paridade par) e, em alguns casos, a decodificação.

Decimal	**Código 2 entre 5**
0	00011
1	00101
2	00110
3	01001
4	01010
5	01100
6	10001
7	10010
8	10100
9	11000

- **Decodificador** (*decoder*) – recebe e detecta uma sequência específica de N bits e ativa uma e apenas uma das 2^N saídas:

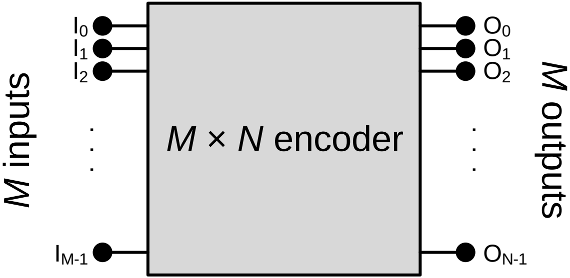
Figura: Decodificador $N \times M$



- **Atenção**: Há decodificadores Ativa-ALTO e Ativa-BAIXO!

- **Codificador** (*encoder*) faz o contrário – uma e apenas uma das 2^N entradas é ativada, gerando a sequência de N bits correspondente:

Figura: Codificador $N \times M$



E se mais de uma entrada estiver ativada, qual a saída?
A solução é o codificador de prioridades.

Exemplo de aplicação para *decoders:*

- Gerar sinais de temporização ou sequenciamento de operações

Exemplo de aplicação para *encoders:*

- Codificador de interruptores/chaves/botões (*switches*) – teclado

Conversores de código – converte a representação dos dados de um formato para outro. *Decoders* só ativam uma saída. Conversores, não necessariamente.

Exemplos de *codecs* ou conversores:

- Binário $\longleftrightarrow$ octal, hexadecimal etc.
- Binário $\longleftrightarrow$ ASCII.
- BCD $\longleftrightarrow$ Decimal
- BCD $\longleftrightarrow$ Display de sete segmentos
- Código Gray $\longleftrightarrow$ Binário
- ASCII $\longleftrightarrow$ EBCDIC
- *et cetera*

Uso de um decodificador binário $\longrightarrow$ ASCII

01010100	01100101	01110011	01110100	01100101
T	e	s	t	e

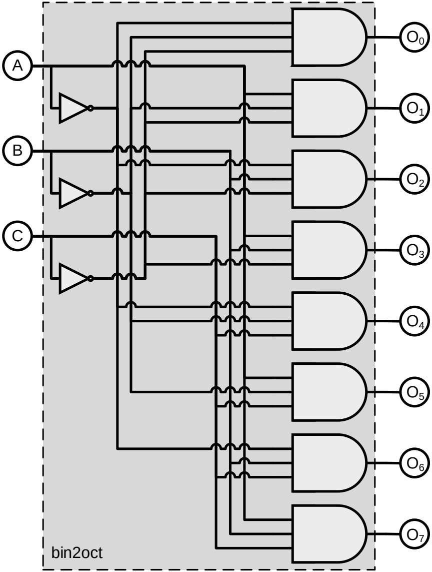
Decoder ativa-alto 3×8 genérico – *aka* decodificador binário para octal (`bin2oct`)

C	B	A	O_0	O_1	O_2	O_3	O_4	O_5	O_6	O_7
0	0	0	1	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0
1	0	1	0	0	0	0	0	1	0	0
1	1	0	0	0	0	0	0	0	1	0
1	1	1	0	0	0	0	0	0	0	1

Como montar o circuito a partir da tabela?

- Sabe-se que tem 3 entradas (`A`, `B` e `C`) e $2^3 = 8$ saídas (O_0 a O_7);
- Monta-se o Mapa K para cada coluna da saída, simplificando o circuito;
- Realizam-se as ligações.

Decoder ativa-alto 3 × 8 genérico – *aka* decodificador binário para octal (`bin2oct`)



Encoder ativa-alto 3 × 8 genérico – *aka* codificador octal para binário (`oct2bin`)

$\mathbf{I_0}$	$\mathbf{I_1}$	$\mathbf{I_2}$	$\mathbf{I_3}$	$\mathbf{I_4}$	$\mathbf{I_5}$	$\mathbf{I_6}$	$\mathbf{I_7}$	**C**	**B**	**A**
1	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	0	0	1	0	0	0	0	0	1	1
0	0	0	0	1	0	0	0	1	0	0
0	0	0	0	0	1	0	0	1	0	1
0	0	0	0	0	0	1	0	1	1	0
0	0	0	0	0	0	0	1	1	1	1

Como montar o circuito a partir da tabela?
Somando as entradas em que a saída é verdadeira:
$\mathbf{C} = \mathbf{I}_4 + \mathbf{I}_5 + \mathbf{I}_6 + \mathbf{I}_7$.
Qual o circuito para o codificador `oct2bin`?

Decoder BCD para decimal (`bcd2dec`)

Tabela: Código BCD (*Binary-Coded Decimal*)

Dec	**BCD (DCBA)**
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

$2^4 = 16$ mas o BCD codifica apenas 10 símbolos diferentes

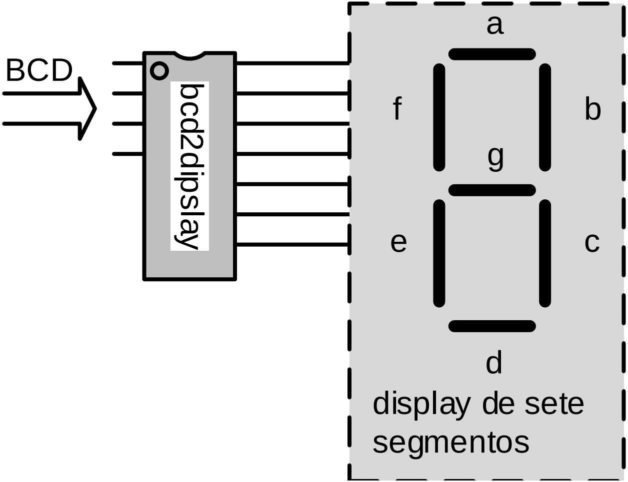
Decoder Octal para Binário (`oct2bin`)

D	C	B	A	O_0	O_1	O_2	O_3	O_4	O_5	O_6	O_7	O_8	O_9
0	0	0	0	1	0	0	0	0	0	0	0	0	0
0	0	0	1	0	1	0	0	0	0	0	0	0	0
0	0	1	0	0	0	1	0	0	0	0	0	0	0
0	0	1	1	0	0	0	1	0	0	0	0	0	0
0	1	0	0	0	0	0	0	1	0	0	0	0	0
0	1	0	1	0	0	0	0	0	1	0	0	0	0
0	1	1	0	0	0	0	0	0	0	1	0	0	0
0	1	1	1	0	0	0	0	0	0	0	1	0	0
1	0	0	0	0	0	0	0	0	0	0	0	1	0
1	0	0	1	0	0	0	0	0	0	0	0	0	1
1	0	1	0										
1	0	1	1										
1	1	0	0										
1	1	0	1										
1	1	1	0										
1	1	1	1										

Qual o circuito para o decodificador `bcd2dec`? Vou fazer apenas as duas primeiras colunas

Conversor/*driver* BCD para display de sete segmentos

Figura: *Layout* básico da ligação do driver



Conversor/*driver* BCD para display de sete segmentos

W	X	Y	Z	a	b	c	d	e	f	g
0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	0	1	1	0	0	0	0
0	0	1	0	1	1	0	1	1	0	1
0	0	1	1	1	1	1	1	0	0	1
0	1	0	0	0	1	1	0	0	1	1
0	1	0	1	1	0	1	1	0	1	1
0	1	1	0	1	0	1	1	1	1	1
0	1	1	1	1	1	1	0	0	0	0
1	0	0	0	1	1	1	1	1	1	1
1	0	0	1	1	1	1	1	0	1	1
1	0	1	0	x	x	x	x	x	x	x
1	0	1	1	x	x	x	x	x	x	x
1	1	0	0	x	x	x	x	x	x	x
1	1	0	1	x	x	x	x	x	x	x
1	1	1	0	x	x	x	x	x	x	x
1	1	1	1	x	x	x	x	x	x	x

x $\rightarrow$ *Don't care*. Qual o circuito? Vou fazer apenas as duas primeiras colunas.

Conversor Gray para binário (gray2bin)

Binário	**Código Gray**
0000	0000
0001	0001
0010	0011
0011	0010
0100	0110
0101	0111
0110	0101
0111	0100
1000	1100
1001	1101
1010	1111
1011	1110
1100	1010
1101	1011
1110	1001
1111	1000

Multiplexador/demultiplexador – MUX/DEMUX

- Multiplexador (mux ou *multiplexer*) $\rightarrow$ circuito seletor/chaveador de dados;
- Apenas um dos diversos sinais de entrada são transferidos para a saída do multiplexador, de acordo com o seletor;
- O demultiplexador (demux ou *demultiplexer*) redireciona o sinal de entrada para uma saída específica de acordo com o seletor.

Diagrama de um multiplexador de quatro entradas:

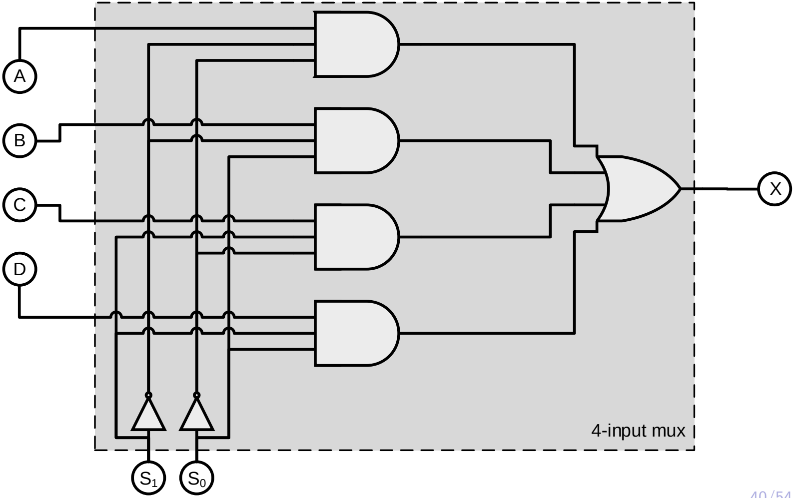
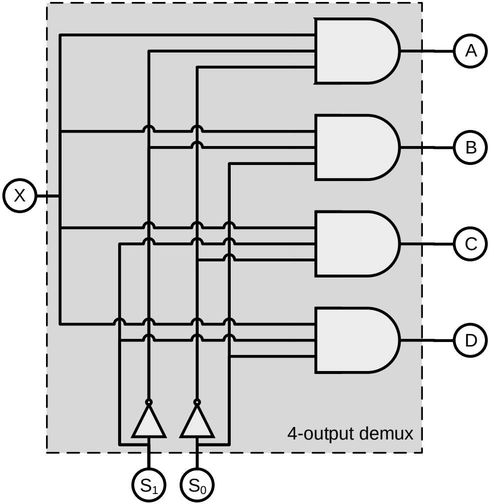
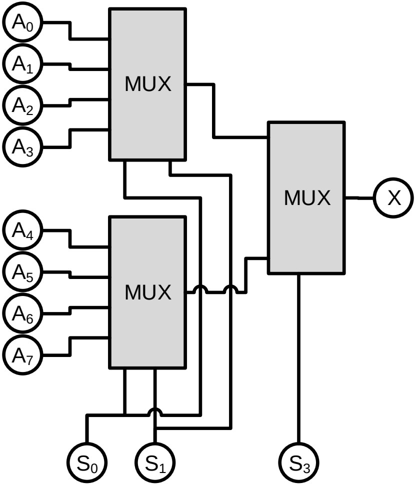


Diagrama de um demultiplexador de quatro saídas:



Multiplexador/demultiplexador – MUX/DEMUX

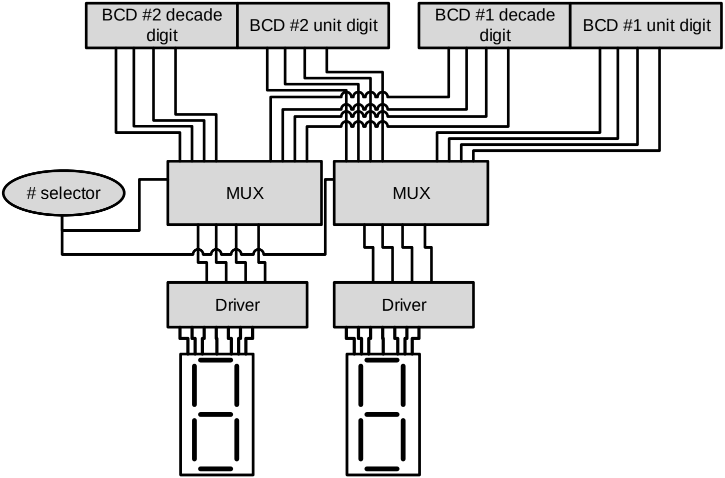
Associação de dois MUXes de quatro entradas e um MUX de duas entradas para formar um de oito entradas:



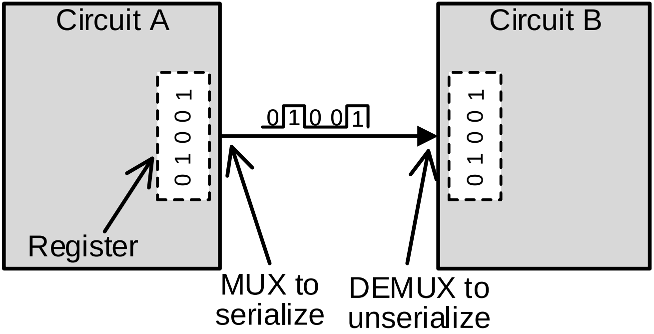
Aplicações:

1. **Roteamento de dados para compartilhamento de circuitos lógicos** – economia de energia, conexões, componentes, tamanho da placa de circuito impresso (PCB)
2. **Conversão de transmissão serial para paralela**;
3. **Conversão de transmissão paralela para serial (UART)**;
4. **Sequenciar operações** – controlar um processo industrial através do sequenciamento de operações;
5. **Implementar circuitos combinacionais** – para reproduzir diretamente uma tabela-verdade desejada (FPGA).

Exemplo de roteamento de dados:

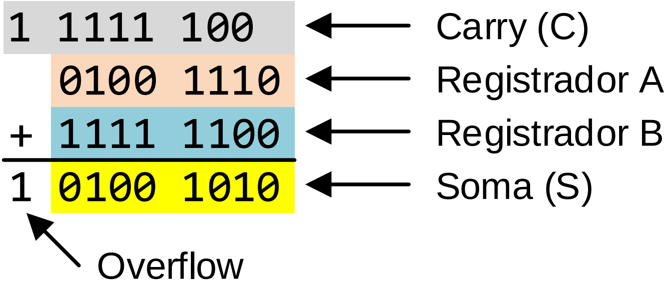


Exemplo de UART – *Universal Asynchronous Receiver/Transmitter*



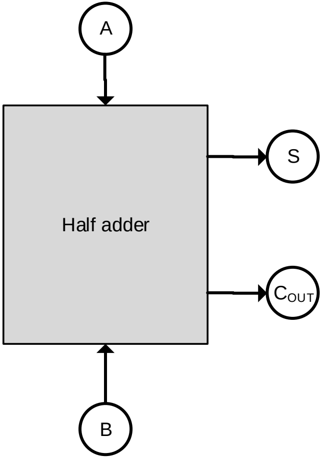
Circuitos aritméticos

- São a base da Unidade Lógico-Aritmética, que opera sobre dados binários em registrador
- Basicamente há dois tipos de somadores binários:
  - Paralelo/completo;
  - Meio somador (*half adder*);
- Antes, devemos recapitular soma binária:



Meio somador

Figura: Diagrama básico de um meio somador



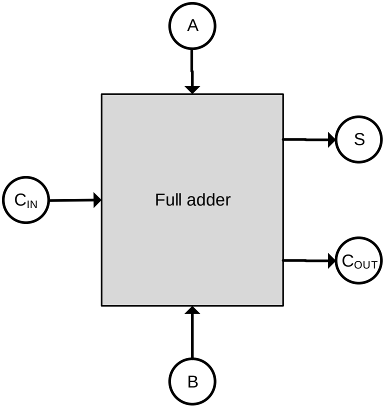
Somador completo

A	**B**	**S**	$\mathbf{C_{out}}$
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Qual o circuito?

Somador completo

Figura: Diagrama básico de um somador completo



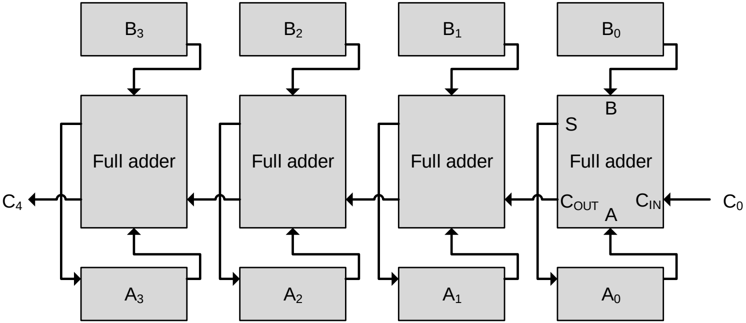
A diferença para o *half adder* é que existe uma entrada para *carry*.

Somador completo

A	B	C_{in}	S	C_{out}
0	0	0		
0	0	1		
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

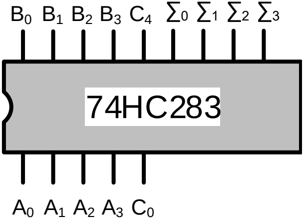
Qual o circuito?

Figura: Diagrama básico de um somador paralelo completo de 4 bits – ligação em cascata



Problema? Propagação do sinal de *carry*! Uma solução: *carry* antecipado.

Figura: Diagrama básico do 74HC283 – somador paralelo integrado de 4 bits



Como fazer um somador de oito bits usando o 74HC283?

- E como fazer um subtrator?
- Como a gente aprendeu a subtrair números em binário?
- Desafio: como seria um circuito somador/subtrator?

Material

Seções utilizadas do livro (11ª edição!):

- 4-1; 4-2; 4-3; 4-4; 4-5; 4-6; 4-7; 4-9.
- 9-1; 9-2; 9-3; 9-4; 9-6; 9-7; 9-8; 9-11.
- 6-9; 6-10; 6-11; 6-12; 6-13; 6-14; 6-15.